Unsupervised Morphological Segmentation Using Neural Word Embeddings

Ahmet Üstün[1] and Burcu Can[2]

[1] Cognitive Science Department, Informatics Institute
Middle East Technical University (ODTÜ)
Ankara, 06800, Turkey
ustun.ahmet@metu.edu.tr
[2] Department of Computer Engineering, Hacettepe University
Beytepe, Ankara, 06800, Turkey
burcucan@cs.hacettepe.edu.tr

Abstract. We present a fully unsupervised method for morphological segmentation. Unlike many morphological segmentation systems, our method is based on semantic features rather than orthographic features. In order to capture word meanings, word embeddings are obtained from a two-level neural network [11]. We compute the semantic similarity between words using the neural word embeddings, which forms our baseline segmentation model. We model morphotactics with a bigram language model based on maximum likelihood estimates by using the initial segmentations from the baseline. Results show that using semantic features helps to improve morphological segmentation especially in agglutinating languages like Turkish. Our method shows competitive performance compared to other unsupervised morphological segmentation systems.

Keywords: morphology, semantics, neural representation of speech and language, morphological segmentation, unsupervised learning, word embeddings

1 Introduction

Morphological analysis is the heart of nearly all natural language processing tasks, such as sentiment analysis, machine translation, information retrieval etc. Such natural language processing tasks become infeasible without any morphological analysis. One reason is the sparsity that is the result of a high number of word forms which introduces out-of-vocabulary (OOV). Morphological segmentation is a way to deal with language sparsity by introducing the common segments within the words rather than dealing with word forms (having multiple morphemes). Hankamer [7] claims that the number of word forms is infinite in agglutinating languages like Turkish.

Morphology is strongly connected to other linguistic levels, such as syntax and semantics. Although this connection has been addressed many times in the

literature, semantic features have been scarcely used in morphological segmentation.

In this paper, we propose an unsupervised method to morphological segmentation that integrates morphotactics with semantics. Our method makes use of semantic similarity between words in order to learn the segmentation points. For example, *book-booking*, *booking-bookings*, *book-booker* are all semantically similar, which gives a clue while segmenting the word *bookings* into *book*, *ing* and *s*; *booker* into *book* and *er*. Using the semantic features, we infer the segmentation points and we use the potential segmentation points in order to model the morphotactic rules for the morpheme transitions. This forms the baseline model in this paper.

We obtain the semantic features of words from the word embeddings, which are learned by a two-layer neural networks [11]. Thus, word meanings are represented in a low-dimensional vector space. We use the cosine similarity between the word embeddings in order to measure the semantic similarity.

Morphotactics of the language is modeled by maximum likelihood estimate based on the initial segmentations of words obtained from the baseline model. Hence, we integrate semantics and morphotactics within the same model.

The paper is organized as follows: Section 2 addresses the related work on unsupervised morphological segmentation, Section 3 describes the mathematical model where Section 3.2 describes our baseline model based on semantic similarity and Section 3.3 describes the bigram model based on maximum likelihood estimates, and Section 4 presents the experiment results compared to other systems.

2 Related Work

Many of the unsupervised morphological segmentation systems are based on word-level orthographic patterns. Goldwater et al. [6] present a two-stage Bayesian model, where morpheme types are drawn from a multinomial distribution (i.e. *generator*) and tokens are generated by a Pitman-Yor process to have a power-law distribution over word frequencies (i.e. *adaptor*).

Morfessor family involves different unsupervised morphological segmentation models. Morfessor Baseline [4] is based on Minimum Description Length (MDL) model and aims to find the lexicon of morphemes that will minimize the corpus length.

Morfessor CatMAP (Creutz and Lagus [5]) performs morphological segmentation using a lexicon of morpheme types and corpus that involves the word tokens to be segmented. Morpheme features such as frequency or perplexity are used as ortographic features. Morphotactics is also modeled with a maximum a posteriori framework (MAP) using Hidden Markov Models (HMMs) for the morpheme transitions. Words are represented as HMMs with four hidden variables: stem, prefix, suffix, and non-morph as for the noisy morphemes. The results show that modeling morphotactics improves morphological segmentation. Our

method is similar to Morfessor in term of handling the morphotactics. This is one of the intuitions that we follow in this paper.

Connection between morphology and syntax has been previously addressed in the literature. Can and Manandhar [1] use syntactic categories obtained by context distribution clustering of Clark [2] for finding morphological paradigms. Lee et al. [9] incorporates part-of-speech information with the morphological segmentation on Arabic. Results show that using syntactic context also improves morphological segmentation.

Connection between morphology and semantics has also been addressed in the literature. Schone and Jurafsky [14] use Latent Semantic Analysis (LSA) to measure the semantic similarity between similar surface forms of the words. This comes from the idea that words which are morphologically derived from each other also semantically similar. For example, *car* and *care* are semantically not similar, therefore they cannot be morphologically derived from each other.

We are also inspired by Schone and Jurafsky [14] in this paper. We learn the segmentation points of the words using the semantic similarity between word forms. Differently from Schone and Jurafsky [14], we use neural word embeddings with a low dimensional vector space to measure semantic similarity.

Narasimhan et al. [12] present a log-linear model, where semantic information is used as feature in the model. Semantic features are obtained from vector space representations and cosine similarity computed between word pairs, which is similar to our model presented in this paper.

Soricut and Och [15] capture morphological rules and morphological paradigms from semantic features obtained from a vector space of words.

Our model resembles the works by Schone and Jurafsky [14] and Narasimhan et al. [12] since we also use the semantic similarity to learn morphological units in words. Progress in learning word embeddings from neural networks have made the semantic information available in computational linguistics in the recent years. Here, we also adopt the semantic information obtained from neural word embeddings in this paper.

3 Morphological Segmentation with Word Embeddings

3.1 Model Overview

Our model consists of three steps that are performed sequentially:

1. We obtain word embeddings from a raw corpus in order to collect semantic information. Skip-gram [11] model is used to build word vectors in 200 dimensional vector space.
2. Words are initially segmented recursively by measuring the semantic similarity between substrings in each word, which forms the baseline model. Semantic similarity is computed by using the word embeddings obtained in the first step.
3. We model the morphotactics by adopting a bigram language model with a maximum likelihood estimate.

Details of three steps are explained below.

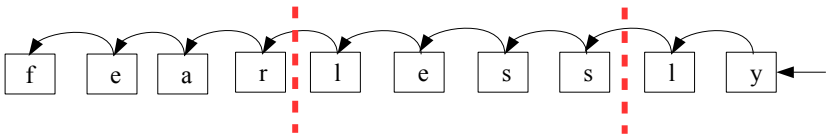

Fig. 1. The illustration that shows the segmentation of the word *fearlessly*. The segmentation is performed from the right end of the word by stripping off letters depending on the manually set semantic similarity.

Table 1. The cosine similarity between the substrings of the word *fearlessly*.

	Word	Remaining substring	Cosine similarity	Segmentation
1	fearlessly	fearlessl	-1	fearlessly
2	fearlessly	fearless	0.34	fearless-ly
3	fearless	fearles	0.14	fearless-ly
4	fearless	fearle	-1	fearless-ly
5	fearless	fear	0.26	fear-less-ly
6	fear	fea	-1	fear-less-ly
7	fear	fe	-1	**fear-less-ly**

3.2 Morphological Segmentation Using Semantic Similarity

Morphological affixation, either inflection or derivation, is strongly connected to semantic derivation that leads to different word forms which are all semantically related with each other. This semantic relation constitutes one side of creativity in language as a cognitive phenomenon. For example, English word *fear* has a close semantic relation with *fearless* and *fearlessly*. The same also holds for inflection. The words *car* and *cars*, *study* and *studied* have got a very close semantic relation.

The initial segmentation is based on this semantic relation between different word forms which are all derived from the same root. We detect the segmentation boundaries based on the semantic similarity between substrings of a word. Semantic similarity is calculated by cosine similarity between word embeddings of the substrings in n dimensional space as follows:

$$\cos(v(w_1), v(w_2)) = \frac{v(w_1) \cdot v_i(w_2)}{\|v(w_1)\| \cdot \|v_i(w_2)\|} \tag{1}$$

$$= \frac{\sum_{i=1}^{n} v_i(w_1) \cdot v_i(w_2)}{\sqrt{\sum_{i=1}^{n} v_i(w_1)^2} \cdot \sqrt{\sum_{i=1}^{n} v_i(w_2)^2}} \tag{2}$$

where $v(w_1)$ denotes the word embedding for of the word w_1, $v(w_2)$ denotes the word embedding for the word w_2 and $v_i(w_1)$ is the ith item in $v_i(w_1)$.

In each iteration of the algorithm, one letter is stripped from the right end of the word. If the cosine similarity between the remaining substring of the word and the substring before stripping a letter is above a threshold value, the segmentation point is accepted. Otherwise, another letter is stripped from the

Algorithm 1 Baseline segmentation algorithm that detects potential morpheme boundaries based on cosine similarity.

```
1: procedure SEMANTIC_PARSING(word)
2:     boundaryList ← ∅
3:     threshold ← d
4:     wordLen ← LEN(word)
5:     charNo ← 1
6:     oldWord ← word
7:     counter ← wordLen
8:     while counter > 1 do
9:         suffix ← LAST_CHAR(charNo, oldWord).
10:        newWord ← FIRST_CHAR(wordLen − charNo, oldWord).
11:        distance ← COSINE(newWord, oldWord)
12:        if distance > threshold then
13:            boundaryList ← PUT(suffix)
14:            oldWord ← newWord
15:            charNo ← 1
16:            wordLen ← LEN(oldWord)
17:        else
18:            charNo ← charNo + 1
19:        counter ← counter − 1
20:    return boundaryList
```

end of the word and concatenated with the letter which has been stripped in the previous iteration. Similarly, a segmentation point is introduced or not depending on the cosine similarity between the remaining substring and the substring with the stripped letters. The entire word is checked from right to left by detecting potential segmentation points until all the letters of the word are stripped. An example segmentation is given in Figure 1 and corresponding cosine similarity values for the same word are given in Table 1.

We use a manually set threshold for the cosine similarity d to decide whether a semantic relation is protected between word forms or not. The entire procedure is given in Algorithm 1. $LEN(word)$ is the number of letters in the word, $LAST\_CHAR(charNo, oldWord)$ returns the rightmost substring that has a number of $charNo$ letters in $oldWord$, $FIRST\_CHAR(charNo, oldWord)$ returns the leftmost substring that has a number of $charNo$ letters in $oldWord$, and $COSINE(newWord, oldWord)$ calculates cosine similarity between $newWord$ and $oldWord$ with respect to Equation 1.

Words that are not found in the corpus do not have a word embedding. If an unseen word is encountered, semantic similarity is assigned -1 and no segmentation is suggested for these words. This is the case in the baseline model and a segmentation is suggested for the unseen words in the maximum likelihood model, which is described in the next section.

3.3 Modeling Morphotactics with ML Estimate

Morphotactics involves a set of rules that define how morphemes can be attached to each other [16]. In agglutinating languages like Turkish, Finnish and Hungarian, concatenation of morphemes plays an important role in morphological generation that builds different words and word forms.

We use maximum likelihood estimation to build a bigram language model for morpheme transitions. Unigram probabilities are used for the first segments, namely the roots, and bigram transition probabilities are used for the suffix ordering. Maximum likelihood model is given as follows:

$$\arg \max_{w=m_0+\cdots+m_N \in W} P(w = m_0 + m_1 + \cdots + m_N) = p(m_0) \prod_{i=1}^{N} p(m_i|m_{i-1}) \quad (3)$$

where w is a word in corpus W and m_i refers to the ith morpheme in w that consists of N morphemes. For both unigram and bigram probabilities we use the maximum likelihood estimates that are obtained from the initial segmentations in the baseline model:

$$p(m_0) = \frac{n(m_0)}{K} \quad (4)$$

where $n(m_0)$ is the frequency of m_0 and K is the total frequency of morphemes in the segmented corpus. The bigram probability is calculated as follows:

$$p(m_i|m_{i-1}) = \frac{n(< m_i, m_{i-1} >)}{M} \quad (5)$$

where $n(< m_i, m_{i-1} >)$ is the frequency of the bigram $< m_i, m_{i-1} >$ and M is the total frequency of the bigram types in the segmented corpus.

An end-of-word symbol is added at the end of each word as a morpheme in order to assign a probability for the last morpheme being the final morpheme of a word. For example, the probability of the last morpheme in *fearlessly$* being ly is computed by $p(\$|ly)$.

While calculating the bigram probabilities, root morpheme is changed to a start symbol in order to remove any dependencies between the root and the first morpheme. For example, the probability of the first morpheme after the root being *less* is computed as follows:

$$p(less|S) = \frac{n(< S, less >)}{M} \quad (6)$$

where $n(< S, less >$ is the frequency of *less* seen as the first morpheme just after the root in the corpus.

In order to find the segmentation points in a word, all possible segmentations are obtained and the segmentation with the maximum likelihood probability regarding Equation 3 is selected as the final segmentation of the word. Viterbi algorithm is used for finding the segmentation having the maximum likelihood. We apply Laplace smoothing with additive number 1 to overcome the sparsity problem.

Table 2. Corpora size for English and Turkish

	English	Turkish
Word Embeddings	129M	361M
Semantic Parsing and ML Estimation [8]	878K	617K
Development [8]	694	763
Test and Evaluation [8]	1050	1760

Table 3. F1-measure on Turkish development set for different cosine similarity threshold values

Threshold (d)	Semantic Parsing (%)	Full Model (%)
0.15	40.51	47.51
0.25	37.42	47.82
0.35	30.16	43.58
0.45	25.14	39.95

4 Experiments & Results

4.1 Data

We used publicly available Morpho Challenge [8] data for training and testing. We tested our model on two languages: English and Turkish. The English dataset consists of 878,034 words and the Turkish dataset consists of 617,298 words with their frequencies in the corpus. The frequency is used for the maximum likelihood estimates of the morphemes in the model.

In order to learn the word embeddings for the words, we used manually collected data both for English and Turkish. The English corpus consists of 129 million word tokens and 218 thousand word types, Turkish corpus consists of 361 million word tokens and 725 thousand word types.

The evaluation was performed on Morpho Challenge [8] gold standard data. Corpora details are given in Table 2. For the evaluation, a number of word pairs are sampled from the results that share at least one common morpheme. For each word pair that indeed share a common morpheme according to the gold standard segmentations, one point is given. Total number of points is divided by the total number of word pairs. This gives the precision. The same is applied for recall by sampling word pairs that share at least one common morpheme from the gold standard segmentations and checked from the results whether they indeed share a common morpheme.

During semantic parsing step in the baseline method, we assign cosine similarity threshold $d = 0.25$ to decide whether two substrings of a word are semantically similar or not. In order to manually assign the threshold value d, we perform both semantic parsing and the full model on the Turkish development corpora [8]. Results are given in Table 3 for different values of the cosine similarity threshold value.

Mikolov's [11] word2vec tool and its open source Java implementation Deeplearning4J [17] are used to obtain vword embeddings for the words.

Table 4. Morpheme segmentation results on Turkish corpora

Model	Precision (%)	Recall (%)	F1-measure (%)
Morfessor CatMap [3]	79.38	31.88	45.49
Full Model	50.70	40.07	44.76
Morpho Chain [12]	69.63	31.73	43.60
Aggressive Comp. [10]	55.51	34.36	42.45
Semantic Parsing	61.82	25.42	36.03
Iterative Comp. [10]	68.69	21.44	32.68
Morfessor Baseline [4]	87.35	18.03	29.89
Nicolas [13]	79.02	19.78	31.64
Base Inference [10]	72.81	16.11	26.38

Table 5. Morpheme segmentation results on English corpora

Model	Precision (%)	Recall (%)	F1-measure (%)
Morfessor Baseline [4]	66.30	41.28	50.88
Semantic Parsing	64.85	37.75	47.72
Full Model	62.79	35.40	45.28
Morfessor CatMap [3]	64.44	34.34	44.81

4.2 Evaluation & Results

We compare our model with Morfessor Baseline [4] and Morfessor Categories MAP [3] since these are well known unsupervised morphological segmentation systems and they perform well in both English and Turkish. However, neither of the models use any semantic information for the segmentation task.

We run Morfessor Baseline, Morfessor CatMAP and our model on the same training and test corpora [8]. Turkish results are given in Table 4 with the other Morpho Challenge 2010 [8] participants. English results are given in Table 5 compared to Morfessor Baseline and Morfessor CatMAP. We also provide our baseline model results that are obtained by adopting only semantic parsing given in section 3.2.

Our model comes the second out of nine systems on Turkish with a F-measure 44.76%. Our baseline results are also promising and outperforms Morfessor Baseline. It is promising to see that using semantic similarity between morphologically related words through only the word embeddings is sufficient for a simple morphological segmentation.

Our model also gives competitive results on English when compared to Morfessor Baseline [4] and Morfessor Categories MAP [3] with a F-measure 45.28%.

One of the common errors arise from highly inflected words, which are prone to over-segmentation in the model. Some morphemes are substrings of other morphemes and therefore there is a semantic relation between these substrings. Therefore, two different morphemes are introduced instead of a single morpheme in these cases. For example, *lar* is a valid morpheme in Turkish; *la* and *r* are also valid morphemes in Turkish. Therefore, the word *kitaplar* is segmented into

Table 6. Correct and incorrect examples of English results

Correct Segmentations	Incorrect segmentations
vouch-safe-d	cen-tr-alize-d
dictator-ial	ni-hil-ist-ic
help-less-ness	su-f-fix-es
rational-ist	ba-ti-ste
express-way	sh-o-gun
flow-chart	el-e-v-ation-s
drum-head-s	im-pe-rsonator-s

Table 7. Correct and incorrect examples of Turkish results

Correct segmentations	Incorrect segmentations
patlıcan-lar-ı	tiy-at-ro-lar-da
su-lar-da-ki	gaze-t-e-ci-ydi
balkon-lar-da	sipari-ş-ler-i-n-iz
parti-si-ne	gelişti-ril-ir-ken
varis-ler-den	anla-ya-mıyo-r-du-m
entari-li-nin	uygu-lama-sı-nda-n
üye-ler-i-dir-ler	veri-tabanları-yla

kitap, *la* and *r* since *kitapla* and *kitaplar* are both valid words and semantically related to each other.

Moreover, it is hard to capture derivational suffixes due to the semantic changes that are introduced by derivation. However, overall success of the baseline method is very promising compared to the methods that work only in the ortographic word level.

Examples of both correct segmentations and incorrect segmentations in English and in Turkish that are obtained by our full model are given in Table 6 and Table 7.

5 Conclusion & Future Work

We present a novel probabilistic model for unsupervised morphological segmentation that adopts the word embeddings in order to measure semantic similarity between word forms which are built from the same root. The results show that incorporating semantic similarity helps finding morphologically related words that leads to find the morpheme boundaries in a simple approach.

We adopt a bigram language model by using maximum likelihood estimate in order to learn morphotactics in the bigram level. We plan to use a better language model for the morpheme transitions in the future.

Acknowledgements

This research was supported by TUBITAK (The Scientific and Technological Research Council of Turkey) grant number 115E464.

References

1. Can, B., Manandhar, S.: Clustering morphological paradigms using syntactic categories. In: Multilingual Information Access Evaluation I. Text Retrieval Experiments: 10th Workshop of the Cross-Language Evaluation Forum, CLEF 2009, Corfu, Greece, September 30 - October 2, 2009, Revised Selected Papers. pp. 641–648. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
2. Clark, A.: Inducing syntactic categories by context distribution clustering. In: Proceedings of the 2Nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7. pp. 91–94. ConLL '00, Association for Computational Linguistics, Stroudsburg, PA, USA (2000)
3. Creutz, M., Lagus, K.: Inducing the morphological lexicon of a natural language from unannotated text. In: Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR 2005. pp. 106–113 (2005)
4. Creutz, M., Lagus, K.: Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0. Technical Report A81 (2005)
5. Creutz, M., Lagus, K.: Unsupervised models for morpheme segmentation and morphology learning. ACM Transactions Speech Language Processing 4, 3:1–3:34 (February 2007)
6. Goldwater, S., Griffiths, T.L., Johnson, M.: Interpolating between types and tokens by estimating power-law generators. In: Advances in Neural Information Processing Systems 18. p. 18 (2006)
7. Hankamer, J.: Finite state morphology and left to right phonology. In: Proceedings of the Fifth West Coast Conference on Formal Linguistics (January 1986)
8. Kurimo, M., Lagus, K., Virpioja, S., Turunen, V.T.: Morpho challenge 2010. http://research.ics.tkk.fi/events/morphochallenge2010/ (June 2011), online; accessed 4-July-2016
9. Lee, Y.K., Haghighi, A., Barzilay, R.: Modeling syntactic context improves morphological segmentation. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learning. pp. 1–9. CoNLL '11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011)
10. Lignos, C.: Learning from unseen data. In: Kurimo, M., Virpioja, S., Turunen, V., Lagus, K. (eds.) Proceedings of the Morpho Challenge 2010 Workshop. pp. 35–38. Aalto University, Espoo, Finland (2010)
11. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013)
12. Narasimhan, K., Barzilay, R., Jaakkola, T.S.: An unsupervised method for uncovering morphological chains. Transactions of the Association for Computational Linguistics (TACL) 3, 157–167 (2015)
13. Nicolas, L., Farré, J., Molinero, M.A.: Unsupervised learning of concatenative morphology based on frequency-related form occurrence. In: Kurimo, M., Virpioja, S., Turunen, V., Lagus, K. (eds.) Proceedings of the Morpho Challenge 2010 Workshop. pp. 39–43. Aalto University, Espoo, Finland (2010)

14. Schone, P., Jurafsky, D.: Knowledge-free induction of inflectional morphologies. In: Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies. pp. 1–9. NAACL '01, Association for Computational Linguistics, Stroudsburg, PA, USA (2001)
15. Soricut, R., Och, F.: Unsupervised morphology induction using word embeddings. In: Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL. pp. 1627–1637 (2015)
16. Sproat, R.W.: Morphology and computation. MIT press (1992)
17. Team, D.D.: Deeplearning4j: Open-source distributed deep learning for the jvm, apache software foundation license 2.0. http://deeplearning4j.org/ (May 2016)